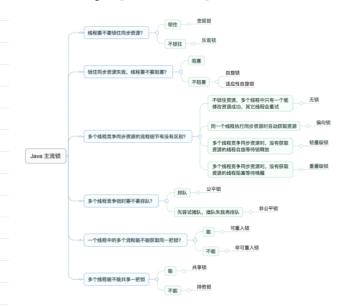
Java并发机制的底层实现原理

2021年10月2日 星期六 14:57

并发三大特性: 0 现性.

四原州生.

③有序性 指令按照预期顺序执行



volatile 欧一保证共享重的一致性. 为前面加了Lock

原理——有volatile的共享变量写完多出一行证编代码

Lock前缀在多核CPU下:

- ①将当前 CPU 缓充行写回内容。
- ②写回内存的操作会使其他CPU里缓存了该内存地址的数据失效.

《我在一致性协议、每个CPU面过总线上传播的数据来检查自己缓忘是否过期。

被用优化。

①采用追加罗节的方式状状出队和入队 为避免头节医和尾结点 Load到的一线中行. Jok7. J.U. J. Linked Transfer Que ue 追加到364岁

原理: CPU高速缓布行是 64岁. 若一个node,不足 64岁. 一一个同节点会读入同个缓旋行. 一) CPU 浏图修设一个node 一一 锁住整个线在行. 一) 台岩在一致性机制 维发一一其他 CPU 无话访问 自己缓冲行 node

色用物景 ① 缓振行为64B的CPU. ②共享变量会被频繁地写

synchronized. D原理。

性. 基础: What: Javan每一个对象都可以作为锁。 普通同步方法。一一钱:当前实例对象。 形式 { 青鹭态同步方法。一一钱:当前案的 Class对象。 同步方法块。一一钱:()内的对象。一定现:monitorenter、monitorexit。

ishovo:

TION 了阿伯罗为明,一般、罗图采取 CONSSTA.
同方法块,一般: ()内的对象,一家规: monitorenter、monitorexit.

where:

Java 对象头。

Class Metadata Address.

Array Length. 数组类型独有 数组长度.

②铁升级和对比 重量级铁。 转量级铁。 /编向铁状态。 无锁状态。

①偏向锐。

对象头中在储 I bit 表示是否为偏向锁,一是 采用 CAS 将对象头的偏向锁指向当前线程、对象头中在储 I bit 表示是否为偏向锁,一 否走 CAS竞争锁。

(少/扁向锁的插旋筒. 等到竞争出现才释放锁的机制.

(2) 关闭偏向锁

-XX:- Use Biased Locking = false. 应用场景: 锁通常情况下属于竞争状态。

②轮量级税.

小加锁

先(AS, 知识则用自旋锁

(2)解锁.

表2-6 锁的优缺点的对比

锁	优 点	缺 点	适用场景
偏向锁	加锁和解锁不需要额外的消耗, 和执	如果线程间存在锁竞争,	适用于只有一个线程访
	行非同步方法相比仅存在纳秒级的差距	会带来额外的锁撤销的消耗	问同步块场景
轻量级锁	竞争的线程不会阻塞,提高了程序的	如果始终得不到锁竞争的	追求响应时间
	响应速度	线程,使用自旋会消耗 CPU	同步块执行速度非常快
重量级锁	线程竞争不使用自旋,不会消耗 CPU	线程阻塞,响应时间缓慢	追求吞吐量
			同步块执行速度较长

原心操作的实视原理。

口术语

缓放行 Cuche Line. 比较新效换 CAS

CPU流域 CPU pipeline. 内在顺序中突 Memory order violation.

缓郁最小操作的

为事方需要于的种周期。 指行解为n步,由cput同航流收线执行,实现一个时钟同期拟行完于指定 由假共享引起.发生时 CPU 必须清空流收线.

多分CPU同时修改同个Cache Line - 其中一个CPU操作无效。

O CPU如何实现原子操作。

(简单 CPU的保证. 成操作) 上九一一十五十二(跨各线宽度

D 如何保证原性 (总线锁定)

四多线锁 Lock并信号。Volotile原理 但一般由线在锁代替。因它把CPU和内在之间的通信锁住了 以缓存锁 保证某个的在地址的操作是锁定的

不会用缓转的情况 (中心方):

③ Java 如何实现原丛操作、〈嵌)(MASCAS:本质为 CPU的 CMPXCHG指令